VERİTABANI YÖNETİM SİSTEMİ SQL SORGU DİLİ ÇALIŞMASI

1. **SELECT Kullanımı**
   Select komutu veri tabanından komut çekmek için kullanılır. Sonuç; sonuç tablosunda tutulur ve buradan çağrılır.
   SELECT * FROM [Database_Name].[Table_Type].[Table_Name] Tablo içinde bulunan tüm sütunları getirir.
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name] Tablo içinden belirli bir sütunu getirir.
   SELECT [Column_Name], [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name] Tablo içinden belirli birden fazla sütunları getirir.
2. **SELECT DISTINCT Kullanımı**
   Bir tabloda, bir sütun birçok tekrarlanan değer içerebilir. Eğer tablodaki farklı değerler listelenmek isteniyorsa Distinct komutu kullanılır.
   SELECT DISTINCT * FROM [Database_Name].[Table_Type].[Table_Name] Tablo içinde bulunan tüm sütunların değerleri arasından farklı olan değerleri tek seferli(yenileme olmadan) getirir.
   SELECT DISTINCT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name] Tablo içinden belirli bir sütun için farklı olan verileri tek seferli(yenileme olmadan) getirir.
   SELECT DISTINCT [Column_Name], [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name] Tablo içinden belirli birden fazla sütunlar için farklı olan verileri tek seferli(yenileme olmadan) getirir.
3. **JOINS Kullanımı**
   Tablolar arasında bazı sütunlar arasındaki ilişkiye dayalı olarak iki ya da daha fazla tablodan veri sorgulamak için kullanılır.
   - **JOIN:** Her iki tabloda en az bir eşleşme olduğunda satır döndürür.
   - **LEFT JOIN:** Sağ tabloda hiçbir eşleşme olmasa bile sol tablodaki tüm satırlar döndürülür.
     SELECT [Column_Name_1], [Column_Name_2]
     FROM [Database_Name].[Table_Type].[Table_Name]
     LETF JOIN [Database_Name].[Table_Type].[Table_Name]
     ON [Database_Name].[Table_Type].[Table_Name]. [Column_Name_1_(Primary Key)] = [Database_Name].[Table_Type].[Table_Name]. [Column_Name_2_(Primary Key)]
   - **RIGHT JOIN:** Sol tabloda hiçbir eşleşme olmasa bile sağ tablodaki tüm satırlar döndürülür.
   - SELECT [Column_Name_1], [Column_Name_2]
     FROM [Database_Name].[Table_Type].[Table_Name]
     RIGHT JOIN [Database_Name].[Table_Type].[Table_Name]
     ON [Database_Name].[Table_Type].[Table_Name]. [Column_Name_1_(Primary Key)] = [Database_Name].[Table_Type].[Table_Name]. [Column_Name_2_(Primary Key)]
   - **INNER JOIN:** İki veya daha fazla tabloda sadece ilişki bulunan satırları listeler.
     SELECT [Column_Name_1], [Column_Name_2]
     FROM [Database_Name].[Table_Type].[Table_Name]
     INNER JOIN [Database_Name].[Table_Type].[Table_Name]
     ON [Database_Name].[Table_Type].[Table_Name]. [Column_Name_1_(Primary Key)] = [Database_Name].[Table_Type].[Table_Name]. [Column_Name_2_(Primary Key)]
   - **FULL OUTER JOIN:** Tabloda ilişkili olmayan satırları da gösterir.
     SELECT [Column_Name_1], [Column_Name_2]
     FROM [Database_Name].[Table_Type].[Table_Name]
     FULL OUTER JOIN [Database_Name].[Table_Type].[Table_Name]
     ON [Database_Name].[Table_Type].[Table_Name]. [Column_Name_1_(Primary Key)] = [Database_Name].[Table_Type].[Table_Name]. [Column_Name_2_(Primary Key)]

4. **ORDER BY Kullanımı**
   ORDER BY sorgunuzun sonucunun sıralı olarak (büyükten küçüğe veya küçükten büyüğe) görüntülenmesi için kullanılabilecek optional bir yan tümcedir.
   ORDER BY : ASC Belirtilen sütunları en küçükten en yükseğe doğru tablodaki değerlerini sıralar.
   SELECT DISTINCT * FROM [Database_Name].[Table_Type].[Table_Name]
   ORDER BY [Table_Name] ASC
   ORDER BY : DESC Belirtilen sütunları en yüksekten en küçüğe doğru tablodaki değerlerini sıralar.
   SELECT DISTINCT * FROM [Database_Name].[Table_Type].[Table_Name]
   ORDER BY [Table_Name] DESC
5. **GROUP BY Kullanımı**
   Belirtilen kolon veya kolonlardaki tüm satırları bir araya toplar ve bu kolonlar üzerinde "aggregate" fonksiyonlarının çalıştırılmasını izin verir.
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name] = 99
   GROUP BY [Column_Name]
6. **HAVING Kullanımı**
   Belirli bir koşula göre grup oluşturmayı sağlar.
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name] = 80
   GROUP BY [Column_Name]
   HAVING [Farklı_Column_Name] < 45
7. **IN Kullanımı**
   Küme üyeliğini sınamak için kullanılır. Yani, bir değerin bir kümeye ait olup olmadığını sınamak için kullanılır.
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name]
   IN (9,10,14)
   IN olmadan aynı görevi yapan farklı bir kullanım tarzı daha vardır WHERE(Süzgeç) OR ekleriyle bağlanarak yapılır.
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name] = 9
   OR [Farklı_Column_Name] = 10
   OR [Farklı_Column_Name] = 14
8. **LIKE Kullanımı**
   Belirttiğiniz tümceye benzerleri bulmak için kullanılır. Tablo sütunlarında yer alan değerlerin benzer olanlarını getirmek için format oluşturmayı sağlar. Aranan hanelerde, bazı basamaklara tanımsız değer girebilmek için **%** işareti kullanılarak 0 ile 9 sayılarından herhangi birini kabul etmesini sağlarız. Aranan haneler tek tırnak içinde yazılır.
   %1 – İlgili sütun değerlerinden son basamağı 1 olanlarını tespit et. (11, 01, 91, 51)
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name] LIKE '%1'
   %1% - İlgili sütun değerlerinden orta basamağı 1 olanları tespit et. (010, 111, 817)
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name] LIKE '%1%'
   1% - İlgili sütun değerlerinden ilk basamağı 1 olanları tespit et. (11,17,10,13)
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name] LIKE '1%'
   % - İlgili sütun değerlerinden tanımsız olanları tespit et. (0,1,2,3,4,5,6,7,8,9)

SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
WHERE [Farklı_Column_Name] LIKE '%'

9. **BETWEEN Kullanımı**
   Belirtilmiş olan iki değer arasındaki verilerin seçilmesi için kullanılır. İki değer aralığını aramak için değerler tek tırnak içine ve AND ekiyle ayrılarak yazılır.
   SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
   WHERE [Farklı_Column_Name] BETWEEN '5' AND '9'
10. **UNION Kullanımı**
    İki veya daha fazla Select durumlu sonuç kümesini birleştirmek için kullanılır. Birleştirmeden önce yazılan sütun adı temel alınarak üzerine birleştirilme işlemi yapar. Bileştirilmek istenilen tablo ilk tablonun değerlerinin üstüne yazılarak gösterir.
    SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
    UNION SELECT [Column_Name] FROM [Database_Name].[Table_Type].[Table_Name]
11. **COUNT Kullanımı**
    Belirli bir sütundaki değerlerin toplam sayısını verir.
    SELECT COUNT(Column_Name) FROM [Database_Name].[Table_Type].[Table_Name]
12. **SUM Kullanımı**
    Nümerik(Sayısal) bir sütunun toplam değerini verir.
    SELECT SUM(Column_Name) FROM [Database_Name].[Table_Type].[Table_Name]
13. **MAX Kullanımı**
    Seçilmiş olan sayısal sütunun en yüksek değerini verir.
    SELECT MAX(Column_Name) FROM [Database_Name].[Table_Type].[Table_Name]
14. **MIN Kullanımı**
    Seçilmiş olan sayısal sütunun en düşük değerini verir.
    SELECT MIN(Column_Name) FROM [Database_Name].[Table_Type].[Table_Name]
15. **AVG Kullanımı**
    Seçilmiş olan sayısal sütunun ortalama değerini verir.
    SELECT AVG(Column_Name) FROM [Database_Name].[Table_Type].[Table_Name]
16. **DATEDIFF Kullanımı**
    İki tarih arasındaki gün sayısını verir. Kodun içinde dd kullanıldığında gün sayısını verir. Ayrıca hafta için wk, saat için hh, dakika için mi, saniye için ss kullanılabilir. Tarihler 'Yıl.Ay.Gün' şeklinde tek tırnak içine girilerek virgül ile ayırılmalıdır.
    - Saniye(SS), İki tarih arasındaki farkı saniye cinsinden hesaplar.
      SELECT DATEDIFF(SS, 'YYYY.MM.DD', 'YYYY.MM.DD')
    - Dakika(MI), İki tarih arasındaki farkı dakika cinsinden hesaplar.
      SELECT DATEDIFF(MI, 'YYYY.MM.DD', 'YYYY.MM.DD')
    - Saat(HH), İki tarih arasındaki farkı saat cinsinden hesaplar.
      SELECT DATEDIFF(HH, 'YYYY.MM.DD', 'YYYY.MM.DD')
    - Gün(DD), İki tarih arasındaki farkı gün cinsinden hesaplar.
      SELECT DATEDIFF(DD, 'YYYY.MM.DD', 'YYYY.MM.DD')
    - Hafta(WK), İki tarih arasındaki farkı hafta cinsinden hesaplar.
      SELECT DATEDIFF(WK, 'YYYY.MM.DD', 'YYYY.MM.DD')
    - Ay(MM), İki tarih arasındaki farkı ay cinsinden hesaplar.
      SELECT DATEDIFF(MM, 'YYYY.MM.DD', 'YYYY.MM.DD')
    - Yıl(YY), İki tarih arasındaki farkı yıl cinsinden hesaplar.
      SELECT DATEDIFF(YY, 'YYYY.MM.DD', 'YYYY.MM.DD')
17. **INSERT Kullanımı**
    Tabloya yeni bir kayıt eklemek için kullanılır. INSERT işlemi INTO eki kullanılarak tablonun adresi tarif edilir ve eklenecek değerler sütun sırası dikkate alınarak VALUES() parametresi içine her değerin tek tırnak içine yazılarak yapılır. Sütun sırasına göre girilen değerlerde boş

bırakılmak istenilen sütun değeri sırasında araya ek virgül(,) işareti eklenerek sırada o alanı boş bırakabiliriz.

1. Kullanımda; sütun adları belirtilmez sadece değerler belirtilir.
   INSERT INTO [Database_Name].[Table_Type].[Table_Name]
   VALUES('Column_Value_1', 'Column_Value_2', 'Column_Value_3')
2. Kullanımda; hem sütun adları hem de değerler belirtilir.
   INSERT INTO [Database_Name].[Table_Type].[Table_Name](Column_Name_1, Column_Name_2, Column_Name_3)
   VALUES('Column_Value_1', 'Column_Value_2', 'Column_Value_3')

18. UPDATE Kullanımı

Kayıt güncelleme için UPDATE cümlesi kullanılır. Güncelleme işlemini yapılacak sütun ve girilecek yeni değer için SET eki kullanılır ve WHERE eki kullanılarak güncellenecek satır yeri tarif edilir.

```
UPDATE [Database_Name].[Table_Type].[Table_Name]
SET Column_Name = 'New_Value'
WHERE Column_Name = 'Column_Value'
```

19. DELETE Kullanımı

Kayıt silmek için DELETE komutu kullanılır. Silinmesi istenilen satır için WHERE eki kullanılarak sütun ve sütunun değeri belirttirilir.

```
DELETE [Database_Name].[Table_Type].[Table_Name]
WHERE Column_Name = 'Column_Value'
```

20. CREATE TABLE Kullanımı

Bu komut veritabanında yeni bir tablo oluşturmak için kullanılır. Yeni tablo oluştururken, tablo ismi yazılır hemen ardından parantez içi açılarak içine tabloda yer alacak her sütün için sütün ve sütun tipi belirtilerek işlem yapılır. Birden fazla sütun için araya virgül(,) işareti konulur.

```
CREATE TABLE Table_Name(Column_Name_1 Column_1_DataType, Column_Name_2 Column_2_ DataType, Column_Name_3 Column_3_ DataType)
```

21. ALTER TABLE Kullanımı

Bu komut var olan bir tabloya sütun eklemek, silmek veya değiştirmek için kullanılır. Yeni sütun eklerken ADD eki ve silerken ise DEL eki kullanılır. ADD ekli kullanılarak sütun eklemede, sütun adı ve sütun tipi yazılarak işlem yapılır.

```
ALTER TABLE Table_Name ADD Column_Name DataType
```

22. DROP TABLE Kullanımı

Bu komut bir tabloyu silmek için kullanılır.

```
DROP TABLE Table_Name
```